An LSTM-CRF Based Approach to Token-Level Metaphor Detection

Malay Pramanick
Dept. of Computer Science
and Engineering
IIT Kharagpur
West Bengal, India - 721302
malay.pramanick@
iitkgp.ac.in

Ashim Gupta
Dept. of Computer Science
and Engineering
IIT Kharagpur
West Bengal, India - 721302
ashimgupta95@
gmail.com

Pabitra Mitra
Dept. of Computer Science
and Engineering
IIT Kharagpur
West Bengal, India - 721302
pabitra@
cse.iitkgp.ernet.in

Abstract

Automatic processing of figurative languages is gaining popularity in NLP community for their ubiquitous nature and increasing volume. In this era of web 2.0, automatic analysis of metaphors is important for their extensive usage. Metaphors are a part of figurative language that compares different concepts, often on a cognitive level. Many approaches have been proposed for automatic detection of metaphors, even using sequential models or neural networks. In this paper, we propose a method for detection of metaphors at the token level using a hybrid model of Bidirectional-LSTM and CRF. We used fewer features, as compared to the previous state-of-the-art sequential model. On experimentation with VUAMC, our method obtained an F-score of 0.674.

1 Introduction

A *metaphor* is a figure of speech that brings together different concepts, which are often distinct and seemingly unrelated. A metaphor comprises a word or a phrase representing something else, where applying it in its literal sense is often not possible. Metaphors bring in vivid imagery to our communications by drawing an analogy between one thing and another or between actions.

Metaphors also provide a fundamental cognitive and structural role. Lakoff and Johnson (1980) introduced metaphor as a central cognitive device that gives structure to abstract conceptual domains, referred to as the 'target domains', which are described in terms of concrete domains, referred to as the 'source domains'. In our work, we do not try to ascertain the source or target domains, rather we focus on determining the presence of metaphorically used tokens in any given sentence.

To estimate the frequency of occurrence of metaphors, Shutova and Teufel (2010) conducted a study on a subset of the British National Corpus (Consortium and others, 2007) and manually annotated the metaphorical expressions in that data. They found out that 241 sentences contained at least one metaphor among the 761 sentences considered.

Figurative uses of language are abundant in literature, but they are not restricted to the literary works. Figurative elements of language, especially sarcasm and metaphor, are common in online product reviews, blogs, articles and posts in social networking sites. With the increasing amount of textual data, the number of metaphorical instances is also increasing. As the application of metaphors is pervasive, their interpretation in non-literal ways is required. To process metaphors automatically, their detection is of foremost importance. Their abundance in any language suggests that their detection would benefit the entire Natural Language Processing (NLP) community, for it would benefit methods like paraphrasing, summarization, machine translation, etc. As of now, most of the state of the art machine translations treat text literally and hence errors creep into the automated translations.

There has been an increasing interest in automated processing of metaphors in the NLP community for their pervasiveness in our communications. To analyze and interpret a metaphor, it has to be identified first. Some of the existing computational models for detection of metaphors use a hierarchical organization of conventional metaphors, or selectional restrictions as provided in lexical resources available or by using word embeddings, or conventional mappings of subject-verb, verb-object, subject-object (Shutova, 2015).

In this paper, we treat the problem of token-level metaphor detection as a sequence tagging problem; and sequence tagging problems, like Parts Of Speech (POS) tagging and Named Entity Recognition (NER), have been long dealt in NLP. We approach token-level metaphor detection, with the help of Long Short-Term Memory (LSTM) and Conditional Random Fields (CRF). We try to identify the metaphors in a running text, irrespective of the type of the metaphor. To observe the effectiveness of our proposed method, we have experimented on VUAMC (Steen et al., 2010b), an open domain text corpus, that has been hand-annotated for metaphors at the token level. Our method obtained the state-of-the-art results as compared to previously reported works on token level metaphor detection.

The rest of the paper is organized as follows. We start in Section 2 by discussing existing literature on metaphor detection which compares to our work in at least one facet and compare these with our methodology. Section 3 discusses the preliminaries. Section 4 presents the motivation behind proposing our method. Section 5 provides information about the dataset used in the experiments and discusses the feature set considered. Section 6 provides the experimental details. Section 7 presents the results of our experiments along with some discussions. Section 8 concludes the paper suggesting possible future works.

2 Related Works

Numerous works have been reported on automated processing of metaphors. Shutova (2015) has made a comprehensive review of computational metaphor identification systems as well as metaphor interpretation systems. Initially, computational approaches to metaphor identification heavily relied on hand-coded knowledge, followed by metaphor identification relying on lexical resources. Recently the NLP community has witnessed a growing interest in statistical and machine learning approaches to metaphor identification. In the following paragraphs, we discuss works done in the past that are related to our approach.

Hovy et al. (2013) presented one of the first approaches to metaphor identification with word vectors. They revisited the idea of selectional preference violation as an indication of metaphorical expression but captured the difference in syntactic relations using dependency trees over words. They used tree kernels, a similarity matrix over tree instances, computed using the number of shared subtrees, to train a Support Vector Machine (Cortes and Vapnik, 1995) (SVM) classifier. To construct the different tree representations, they considered word vector, lemma, POS tag, dependency label, and WordNet (Fellbaum, 1998) supersense representations. They downloaded a list of 329 examples of metaphorical expressions from the web and used 80% as training data, 10% as developmental set and remaining 10% as test set. The authors reported an F-score of 0.75, which indicates the importance of syntactic information and compositionality in metaphor identification.

Haagsma and Bjerva (2016) worked on detecting novel metaphors using selectional preference information. They claim that "metaphor is defined by basicness of meaning and not frequency of meaning". Though the basicness and frequency are correlated, there are instances where the figurative sense of a word has become more frequent than its original literal sense. They proposed different ways for generalizing over selectional preferences obtained from a corpus. One among them was to use the word embeddings for the generalizations directly. They used a neural network with one hidden layer containing 600 hidden units with a sigmoid activation function and the resulting predictions were used as the Predicted Log-Probability (P-LP) feature. They evaluated the approaches on the VU Amsterdam Metaphor Corpus (VUAMC).

Tsvetkov et al. (2014) used logistics regression with word vectors and MRC Psycholinguistic Database to get the abstractness and imageability scores. With the abstractness and imageability scores, they used supersenses and vector representation of words as features for Random Forest Classifier to detect metaphor.

Klebanov et al. (2014) considered each of the 'content-word' token in any given text to be classified as metaphorical or not. They used the logistic regression classifier to detect metaphor using unigrams, part of speech, concreteness and topic models as features. Klebanov et al. (2015) tuned the weight parameter to represent concreteness of information and include the difference of concreteness between an adjective and its head noun and between a verb and its direct object, to improve on their previous work.

Do Dinh and Gurevych (2016) presented a neural network based method to detect metaphors at the token level. Their method relied on word embeddings. They experimented with "multilayer perceptrons (MLP), fully connected feedforward neural networks with an input layer, one or more hidden layers, and an output layer". In their experiments, they incorporated labels for tokens with noun, verb, adjective, adverb POS tags as supplied with the VUAMC, as their interest lied in the detection of metaphoricity of content tokens. They also filtered out auxiliary verbs, having lemmas *have*, *be*, or *do*.

Rai et al. (2016) used Conditional Random Fields (CRF) to detect metaphors in an open domain text. For their experiments, they used Syntactic features, Conceptual features, Affective Features and Contextual features. *Lemma, Part of Speech (PoS), Named Entity (NE) type, dependency, and stop word* as a set of syntactic features extracted by using Stanford CoreNLP formed the Syntactic features. *Concreteness, familiarity, imageability, frequency and meaningfulness* extracted from MRC Psycholinguistic Database formed the Conceptual features. *Cognitive state, physical state, trait, attitude, and emotion* extracted from WordNet Affect (Strapparava et al., 2004) formed the Affective features. As Contextual features, they used word embeddings. Using CRF++ (Kudo, 2005) on VUAMC, they reported an F-score of 0.6093.

Do Dinh and Gurevych (2016) filtered out tokens if they did not have noun, verb, adjective or adverb as part of speech. On the other hand, we considered all tokens of the dataset. The reason being that if one word cannot be used metaphorically, it can indicate metaphoricity of another. We used LSTM, which they had suggested in their conclusion. Our approach uses less number of features as compared to that of Rai et al. (2016). We used a hybrid architecture of Bidirectional-LSTM and CRF for metaphor detection.

3 Preliminaries

3.1 Word Embeddings

There is a long history of word embeddings (Hinton et al., 1985; Hinton et al., 1986; Elman, 1990). Collobert and Weston (2008) tried to define a unified architecture for Natural Language Processing. The architecture deals with raw words and transforms them into real-valued vectors. The architecture learns feature representations that have relevance to many well known NLP tasks like part-of-speech (POS) tagging, chunking, named-entity recognition (NER), learning a language model, recognizing synonyms and semantic role-labeling (SRL), by training a deep neural network.

The word embeddings produced by the method of Turian et al. (2010), are real numbers that are not necessarily in a bounded range, however, generally, the embeddings have a zero mean, though they can be scaled by a hyper-parameter to control their standard deviation.

Mnih and Hinton (2009) used a log-bilinear model as the foundation to their hierarchical model. They were focussed on a learning approach where no expert knowledge was available. The 'word feature vectors' were obtained by generating a random tree of words, training a hierarchical log-bilinear model on it and using the distributed representations the model learns while building the tree of words.

Mikolov et al. (2013b) showed that subsampling of frequent words during the training speeds-up the process, and also improves the accuracy of the vector representations of less frequent words. The most common words are usually less informative as they can easily occur millions of times. To counter the rare and common words imbalance, they used a sub-sampling approach. The work provides a simple but powerful way to represent large pieces of text, keeping the computational complexity to a minimal.

Pennington et al. (2014) explicitly made the model properties that were needed for semantic and syntactic regularities and presented a global log-bilinear model having the advantages of global matrix factorization as well as local context window methods.

3.2 LSTM

Long Short-Term Memory (LSTM) was introduced by Hochreiter and Schmidhuber (1997) to overcome the issue of vanishing gradients in the vanilla recurrent neural networks. They introduced the gating mechanism through LSTM, which made it possible to learn long-term dependencies.

LSTM equations are as follows:

$$\begin{aligned}
i_t &= \sigma(\mathcal{W}_{xi} \cdot \mathcal{X}_t + \mathcal{W}_{hi} \cdot \mathcal{H}_{t-1} \\
&\qquad + \mathcal{W}_{ci} \cdot \mathcal{C}_{t-1} + b_i) \\
f_t &= \sigma(\mathcal{W}_{xf} \cdot \mathcal{X}_t + \mathcal{W}_{hf} \cdot \mathcal{H}_{t-1} \\
&\qquad + \mathcal{W}_{cf} \cdot \mathcal{C}_{t-1} + b_f) \\
C_t &= f_t \odot C_{t-1} + i_t \odot tanh(W_{xc} \cdot \mathcal{X}_t \\
&\qquad + \mathcal{W}_{hc} \cdot \mathcal{H}_{t-1} + b_c) \\
o_t &= \sigma(\mathcal{W}_{xo} \cdot \mathcal{X}_t + \mathcal{W}_{ho} \cdot \mathcal{H}_{t-1} \\
&\qquad + \mathcal{W}_{co} \cdot \mathcal{C}_t + b_o) \\
\mathcal{H}_t &= o_t \odot tanh(C_t)
\end{aligned} \tag{1}$$

In Eq. 1 for the LSTM, σ is the sigmoid function, $\odot$ is the Hadamard product, C_t is the cell state, H_t is the hidden state. i_t, f_t, o_t refer to the input gate, forget gate and output gate respectively.

A Bidirectional-LSTM (Graves and Schmidhuber, 2005) has two LSTM networks. One of the networks is provided the input in the forward direction, whereas the other network is provided the input backward, but both of the networks are connected to the same output layer. In this paper, *Bidirectional-LSTM* is henceforth referred to as Bi-LSTM.

3.3 CRF

While predicting the output tags for a sequence, a system can also make use of the tags predicted in the previous time steps. This can be facilitated by using a Maximum Entropy Markov Model (MEMM) (McCallum et al., 2000) or a Conditional Random Fields based tagging scheme. Conditional Random Fields or CRF was introduced by Lafferty et al. (2001) for building probabilistic models for labeling sequential data. CRF overcomes the problem of label bias. In most problems, CRF provides a better tagging performance as compared to MEMMs (Lafferty et al., 2001; Rozenfeld et al., 2006).

4 Motivation

A standalone word, or token for that matter, cannot be marked for metaphoricity as many words can be used both literally or figuratively, which is determined by the context of the word. Many computational methods have been proposed for metaphor detection in datasets consisting of word tuples like Adjective-Noun (Tsvetkov et al., 2014; Shutova et al., 2016), Noun-Noun (or Type I metaphor as categorised by Krishnakumaran and Zhu (2007)) (Su et al., 2017; Kesarwani et al., 2017) and Subject-Verb-Object (Tsvetkov et al., 2014; Shutova et al., 2016).

Open domain texts may have more than one type of metaphor and though dependency parsing is pretty accurate these days, metaphorically related words and their indication might not be directly related. So inherently detection of metaphors, at a token level, is a context-sensitive job and a sequential one.

Hybrid models of Bidirectional-LSTM and CRF have been successful in tagging problems like POS tagging, chunking and NER tagging (Huang et al., 2015; Lample et al., 2016). We apply a hybrid model of Bidirectional-LSTM and CRF (henceforth referred to as **Bi-LSTM-CRF**), to look for metaphors at the token level.

5 Data and Feature Set

5.1 Dataset

VU Amsterdam Metaphor Corpus (VUAMC) (Steen et al., 2010b) is a subset of BNC Baby. The Reference Guide to BNC Baby (2003) describes its design and provides information about the way in which it is encoded. VUAMC is one of the "largest available corpus hand-annotated for all metaphorical language use, regardless of lexical field or source domain". It was reported that the corpus was annotated with an inter-annotator reliability in terms of Fleiss' Kappa, $\kappa > 0.8$.

VUAMC consists of randomly selected texts from four registers of the BNC-Baby, namely, **academic texts**, **conversations**, **fiction** and **news texts**. The texts are coded for metaphor. The annotation manual for VUAMC and a detailed documentation of the project have been published in Steen et al. (2010a).

In VUAMC, each lexical unit is annotated as being used literally or metaphorically. Annotation for metaphoricity is done using fine grained tags. XML tags with attribute **function** having value **mrw** indicates that the unit is related to metaphors (mwr expands to metaphor-related words), but they are further divided with the help of attribute **type** which has values between **bridge**, **lit** and **met**. We considered tags with the value of **met** for attribute **type** when attribute **function** has value of **mrw** as metaphorical and label everything else as literal.

5.2 Generating Word Representations

We obtained word embeddings for our experiments by using the open source Google word2vec[1] (Mikolov et al., 2013a; Mikolov et al., 2013b; Mikolov et al., 2013c). We have used the Continuous Bag-Of-Words (CBOW) model of Mikolov et al. (2013a) with a window size of eight (8) words. CBOW uses a continuous distributed representation of the context but the order of words in the history does not influence the projection.

For training the model, we used the text corpus from recent English Wikipedia dump[2] preprocessed with the Perl script of Matt Mahoney[3] and obtained vectors with a dimension of 200.

By training the model with Wikipedia text corpus, we obtained word embeddings for most of the lemmas and words contained in the VUAMC. For some of the words and some of the lemmas, embeddings were not available. There were some words which were compositions of more than one word, for them we took the component-wise average of the vectors of the composing words. Averaging retains the property of both of the components. Phrase embedding could have been an alternative, but averaging sufficed our purpose. Numerical tokens of VUAMC had to be dealt separately as the Perl script removes non-alphabetical characters from the corpus during the preprocessing. So years were represented by the embedding of the word 'year', amount was represented by that of 'dollars', component-wise averaged with embedding for 'million' or 'billion' if mentioned in the token, and so on. For the words whose representations were still not available, a constant vector was used.

In XML file of the VUAMC, the Part-Of-Speech (POS) for the tokens are provided by the "type" attribute. For our experiments, we needed the vector representations of the POS. For their representations instead of using one-hot encoding or some randomly initialized vectors, we trained Google word2vec only on the sequence of POS tags as present in the VUAMC and used the CBOW model to generate vectors of dimension 20 for the POS. While training word2vec on the sequence of POS tags, we did not include the labels for metaphoricity, keeping the embedding generation for the POS unsupervised.

[1]https://code.google.com/archive/p/word2vec/
[2]https://dumps.wikimedia.org/enwiki/latest/
[3]http://mattmahoney.net/dc/textdata.html

5.3 Features

The features that we considered for our experiments are as follows :

1. Token
2. Lemma of the token
3. Part-Of-Speech (POS)
4. Whether the lemma and the word are same
5. Whether the lemma is present in the token

Token or word (converted to lower case, if not originally in the XML file of VUAMC) was the most essential component for the feature vector as we were addressing the problem of token-level metaphor detection. So for every experiment performed for this paper, the token was common. The word embedding of the token as generated in subsection 5.2 was considered as a part of the feature vector, and referred to as 'Token'.

Similarly, for the lemma of the token as provided by the "lemma" attribute in XML file of VUAMC, word embeddings as generated in subsection 5.2 was considered and referred to as 'Lemma' in later sections. The generated POS embeddings were used to represent the Part-Of-Speech as provided by the "type" attribute in XML file of VUAMC and referred to as 'POS'.

For the features 4 and 5, we have used one hot encoding. For each of them, there were only two possible scenarios, yes and no, so vectors of dimension 2 did the work. Features 4 and 5 represent the relation between the lemma and the token, so collectively they are referred to as 'Word-Lemma Relations'.

The feature vector of a token, as input to the model, was a concatenation of the representation of the features described above in the order they have been mentioned. When we experimented for the contribution of each of the features over the token, we omitted some features while retaining the others, but we maintained the order for our ease.

6 Experiments

6.1 Baselines

As one of our baselines, we used the results from Do Dinh and Gurevych (2016). Using neural network, they experimented on each of the contained genres in VUAMC (news, conversation, fiction, academic) separately; for each subcorpora, they used a random subset of 76% of the data as a training set, 12% as development set and 12% as test set. They also reported the performance of their system on the complete corpus, with a 76%, 12%, 12% split. We compared with their precision, recall and F1-measure regarding metaphorically used tokens for their tuned neural network on a feature set of **Token+POS+Conc** i.e. with a feature set consisting of **Token**, **POS** and **Concreteness rating**.

As for our other baseline, we considered the results from Rai et al. (2016), as reported by them. They used conditional random fields (CRF) for detection of metaphors and experimented on each of the genres contained in VUAMC, as well as on the complete dataset. For the genres, they have reported precision and recall (for metaphor class), from which we can calculate the F-measure for the metaphor class. On the complete dataset, they have reported precision, recall and F-measure, with which we compared the performance of our method.

6.2 Experimental Setup

We considered all tokens, irrespective of their POS tag supplied with the VUAMC. We ignored the punctuations like comma (**,**), exclamation mark (**!**), period (**.**), and quotation mark (**'**), as punctuation marks cannot be used metaphorically, to the best of our knowledge.

For each of the tokens considered, the feature vector was computed as described in section 5. As the punctuation marks were not considered, the tokens belonging to a particular sentence were clubbed together, in the order they appear in the sentence in VUAMC. As the label for metaphoricity, each token is marked as negative or positive representing **literal** and **metaphorical** tokens, respectively.

As sentences of the dataset are not of equal length, we padded them with constant vectors, labeled negative for metaphoricity. In a running text, if the end of sentences are not marked, an automatic processor for sentences can be used to mark them.

We used a Bi-LSTM-CRF architecture similar to the ones presented by Collobert et al. (2011), Huang et al. (2015) and Lample et al. (2016). Our architecture used a Bidirectional-LSTM with a layer of CRF above it.

Our model with back-propagation updated parameters with every batch. We used a batch size of 128 while training. We used a learning rate of 0.0005 and had set the gradient clipping to 5. We used Adam (Kingma and Ba, 2014) as our learning method with a dropout of 0.5. Our model used a single LSTM layer for forward and a single LSTM layer for backward propagations. Each of the layers had a dimension of 100. It was observed that changing the dimensions did not significantly improve the results.

The system is trained and tested on the complete corpus, leaving out the metadata of the genre they belong to in the British National Corpus (BNC). We did a 10-fold cross validation on the entire dataset, with the order of the sentences changed randomly. We rearranged the sentences so that the sentences belonging to the same genre did not necessarily get clubbed together as originally in the dataset. The performance of the system with the suggested features is evaluated on the basis of Precision, Recall and F1-score.

To check whether a feature contributes to the results, we also experimented on an incremental basis, i.e. adding features on top of the others. We also checked separately for the features along with the word embeddings for the words (tokens). We did this with a 10-fold cross-validation.

6.3 Fig-Lang18 Shared Task

The shared task on metaphor detection in the First Workshop on Figurative Language Processing[4], co-located with NAACL 2018 targets detecting "all content-word metaphors in a given text". The shared task also uses the VUAMC dataset (referred to as VUA in the shared task). It has a separate evaluation only for the verb metaphors.

The training as well as the test data consists of text ids and sentence ids along with the respective sentences from the VUAMC. The test phase has test instances (one set of instances for all-POS and another only for the verb metaphors), over which the submitted predictions are evaluated.

For our training and testing purpose, we had the text ids and sentence ids as provided for the shared task, from which we could get the respective sentences from the VUAMC and thus generate the feature vectors for each of their tokens (leaving aside the punctuation marks), as described in

[4]https://competitions.codalab.org/competitions/17805

Method	Precision	Recall	F_1-score
Do Dinh and Gurevych (2016)	0.5899	0.5355	0.5614
Rai et al. (2016)	0.6333	0.5871	0.6093
Bi-LSTM-CRF (Embeddings only for tokens)	0.7036	0.5755	0.6327
Bi-LSTM-CRF (All of the considered features)	0.7283	0.6253	0.6740

Table 1: Results for complete VU Amsterdam Metaphor Corpus.

Method	Precision	Recall	F_1-score
Only Token	0.7036	0.5755	0.6327
Token + Word-Lemma Relations	0.7040	0.5876	0.6330
Token + POS	0.7252	0.5784	0.6399
Token + Lemma	0.7495	0.6213	0.6657
Token + Lemma + POS	0.7239	0.6297	0.6729
Token + Lemma + POS + Word-Lemma Relations	0.7283	0.6253	0.6740

Table 2: Results for Feature Selection on the complete VU Amsterdam Metaphor Corpus with Bi-LSTM-CRF.

section 5. If any punctuation mark was to be evaluated, it was to be given a negative level for metaphoricity.

We trained on the training set as decided for the task, using the same system of Bi-LSTM-CRF as used in the previous subsection, with all of the features considered. We did not train separately for verb metaphors but used the same system to evaluate the verb metaphors also.

7 Results and Discussions

Using Bi-LSTM-CRF only with the word embeddings of the tokens of the sentences, gives better results as compared to the baselines, as shown in Table 1.

We have also reported the results of experiments for feature selection in Table 2. As it can be seen in Table 2, using word embeddings of the lemmas along with the tokens, improved the results by a huge scale. Adding embeddings for the POS also improved the results. POS tags are provided with VUAMC, but for a dataset, if the POS are not available, they can be generated by using the available POS taggers.

Do Dinh and Gurevych (2016) and Rai et al. (2016) used concreteness ratings but for our method, the results hardly change if we consider concreteness ratings. As Do Dinh and Gurevych (2016) have pointed out, this could be due to *one-dimensionality* of the abstractness (or concreteness) feature.

The results of the experiments on the shared task data have been reported in Table 3. Our method obtained an F-measure of 0.6541 over the entire test set of the shared task but an F-measure of 0.5362 for the all-POS instances and 0.5859 for the verb instances.

8 Conclusion and Future Work

We presented a method for token level metaphor detection using Bi-LSTM-CRF. Our method uses word-embeddings of the token as well as its lemmatized form. Our method compares well with the state-of-the-art system that considers a huge set of features, which we beat with fewer features without filtering out any particular type of word.

The context that we had considered for our experiments was one sentence at a time, but an indication of metaphorically related words can also be across sentences and for those scenarios, the global context is expected to help. So in our future work, we intend to take wider context into consideration.

Acknowledgments

We would like to thank Sonam Singh, Priyanka Sinha and António Anastásio Bruto da Costa for their valuable feedback on the initial draft of this paper. We would also like to thank the anonymous reviewers for their valuable comments and feedback.

Data	**Accuracy**	**Precision**	**Recall**	**F-measure**
All POS Instances	0.8575	0.6446	0.4591	0.5362
Verb Instances	0.7807	0.6753	0.5173	0.5859
Overall Test Set	0.9172	0.7331	0.5904	0.6541

Table 3: Results on Shared Task.

References

Lou Burnard. 2003. Reference guide for bnc-baby.

Ronan Collobert and Jason Weston. 2008. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th international conference on Machine learning*, pages 160–167. ACM.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12(Aug):2493–2537.

British National Corpus Consortium et al. 2007. British national corpus version 3 (bnc xml edition). *Distributed by Oxford University Computing Services on behalf of the BNC Consortium. Retrieved February*, 13:2012.

Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning*, 20(3):273–297.

Erik-Lân Do Dinh and Iryna Gurevych. 2016. Token-level metaphor detection using neural networks. In *Proceedings of the fourth workshop on metaphor in NLP*, pages 28–33.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Christiane Fellbaum. 1998. *WordNet*. Wiley Online Library.

Yulia Tsvetkov Leonid Boytsov Anatole Gershman and Eric Nyberg Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional lstm networks. In *Neural Networks, 2005. IJCNN'05. Proceedings. 2005 IEEE International Joint Conference on*, volume 4, pages 2047–2052. IEEE.

Hessel Haagsma and Johannes Bjerva. 2016. Detecting novel metaphor using selectional preference information. In *Proceedings of the fourth workshop on metaphor in NLP*, pages 10–17.

GE Hinton, DE Rumelhart, and RJ Williams. 1985. Learning internal representations by back-propagating errors. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*, 1.

Geoffrey E Hinton, James L Mcclelland, and David E Rumelhart. 1986. Distributed representations, parallel distributed processing: explorations in the microstructure of cognition, vol. 1: foundations.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Dirk Hovy, Shashank Shrivastava, Sujay Kumar Jauhar, Mrinmaya Sachan, Kartik Goyal, Huying Li, Whitney Sanders, and Eduard Hovy. 2013. Identifying metaphorical word use with tree kernels. In *Proceedings of the First Workshop on Metaphor in NLP*, pages 52–57.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Vaibhav Kesarwani, Diana Inkpen, Stan Szpakowicz, and Chris Tanasescu. 2017. Metaphor detection in a poetry corpus. In *Proceedings of the Joint SIGHUM Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature*, pages 1–9.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Beata Beigman Klebanov, Chee Wee Leong, Michael Heilman, and Michael Flor. 2014. Different texts, same metaphors: Unigrams and beyond. In *Proceedings of the Second Workshop on Metaphor in NLP*, pages 11–17.

Beata Beigman Klebanov, Chee Wee Leong, and Michael Flor. 2015. Supervised word-level metaphor detection: Experiments with concreteness and reweighting of examples. In *Proceedings of the Third Workshop on Metaphor in NLP*, pages 11–20.

Saisuresh Krishnakumaran and Xiaojin Zhu. 2007. Hunting elusive metaphors using lexical resources. In *Proceedings of the Workshop on Computational approaches to Figurative Language*, pages 13–20. Association for Computational Linguistics.

Taku Kudo. 2005. Crf++: Yet another crf toolkit. *Software available at http://crfpp. sourceforge. net*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data.

George Lakoff and Mark Johnson. 1980. *Metaphors we live by*. University of Chicago press.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Andrew McCallum, Dayne Freitag, and Fernando CN Pereira. 2000. Maximum entropy markov models for information extraction and segmentation. In *Icml*, volume 17, pages 591–598.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013a. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013c. Linguistic regularities in continuous space word representations. In *hlt-Naacl*, volume 13, pages 746–751.

Andriy Mnih and Geoffrey E Hinton. 2009. A scalable hierarchical distributed language model. In *Advances in neural information processing systems*, pages 1081–1088.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Sunny Rai, Shampa Chakraverty, and Devendra K Tayal. 2016. Supervised metaphor detection using conditional random fields. In *Proceedings of the Fourth Workshop on Metaphor in NLP*, pages 18–27.

Binyamin Rozenfeld, Ronen Feldman, and Moshe Fresko. 2006. A systematic cross-comparison of sequence classifiers. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 564–568. SIAM.

Ekaterina Shutova and Simone Teufel. 2010. Metaphor corpus annotated for source-target domain mappings. In *LREC*, volume 2, pages 2–2.

Ekaterina Shutova, Douwe Kiela, and Jean Maillard. 2016. Black holes and white rabbits: Metaphor identification with visual features. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 160–170.

Ekaterina Shutova. 2015. Design and evaluation of metaphor processing systems. *Computational Linguistics*, 41(4):579–623.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna Kaal, Tina Krennmayr, and Trijntje Pasma. 2010a. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing.

Gerard J Steen, Aletta G Dorst, J Berenike Herrmann, Anna A Kaal, and Tina Krennmayr. 2010b. Vu amsterdam metaphor corpus.

Carlo Strapparava, Alessandro Valitutti, et al. 2004. Wordnet affect: an affective extension of wordnet. In *LREC*, volume 4, pages 1083–1086.

Chang Su, Shuman Huang, and Yijiang Chen. 2017. Automatic detection and interpretation of nominal metaphor based on the theory of meaning. *Neurocomputing*, 219:300–311.

Yulia Tsvetkov, Leonid Boytsov, Anatole Gershman, Eric Nyberg, and Chris Dyer. 2014. Metaphor detection with cross-lingual model transfer. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 248–258.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: a simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394. Association for Computational Linguistics.